Penerapan Algoritma MTD(f) pada Permainan *TIC TAC TOE* Dengan Variasi Area

Andrian Chanhar[*1], Kelvin Sunanto Husin[2], Renni Angreni[3]
[123]STMIK GI MDP, Jalan Rajawali No.14 Palembang, 0711-376400
e-mail: [*1]andrianchanhar@mhs.mdp.ac.id, [2]kelvinhusin@gmail.com, [3]renni@mdp.ac.id

Abstrak

Game edukasi merupakan salah satu jenis permainan yang memiliki alur untuk mengasah kemampuan dan konsentrasi. Disini penulis menerapkan algoritma Memory-Enhanced Test Driver With Value F pada game Tic Tac Toe. Salah satu tujuan penulis adalah untuk mengetahui cara kerja algoritma Memory-Enhanced Test Driver With Value F serta penerapannya pada Aritficial Intelligent (AI) khususnya pada permainan Tic Tac Toe dengan variasi area. Dari pengujian yang telah penulis lakukan, penulis mendapatkan hasil sesuai dengan yang diharapkan. Adapun Proses pengembangan game ini diterapkan pada platform android yang memungkinkan pengguna dapat bermain dimanapun melalui smartphone. Sehingga dapat ditarik kesimpulan bahwa penerapan algoritma Memory-Enhanced Test Driver With Value F sangat cocok pada permainan Tic Tac Toe karena memudahkan Artificial Intelligent (AI) dalam menentukan langkah terbaik.

Kata kunci*–Game* Game, Tic Tac Toe, Algoritma *Memory-Enhanced Test Driver With Value F*, Phonegap.

Abstract

Education game is one of the games with brain's concentration and ability oriented plot. In this "Tic Tac Toe" game, the writer applied Memory-enchanted Test Driver with Value F's algorithm. One of the writer's purpose was to understand the ways Memory-enchanted Test Driver with Value F work and it's appliance on Artificial Inteligent (AI) , especially on the Area-varieted Tic Tac Toe. The result from the tests that have been done is just as the writer expect. This game's development was applied on android platform that allows the players to play through smartphone wherever they are. The conclusion from the explanation above is that the appliance of Memory-enchated test driver with Value F matches the Tic Tac Toe game because it ease the Artificial Inteligent (AI) at choosing the best way.

Keywords*–Game, Tic Tac Toe, Algorithm Memory-Enhanced Test Driver With Value F, Phonegap.*

1. PENDAHULUAN

Perkembangan *game* edukasi yang inovatif dan variasi harus didukung dengan proses permainan dan manfaat dari *game* itu sendiri agar pada saat *game* dimainkan dapat memberikan efek ketertarikan terhadap *game* tersebut. Salah satu *game* yang memiliki tingkat variasi dan membutuhkan konsentrasi adalah *game Tic Tac Toe*. *Game Tic Tac Toe* merupakan permainan klasik berjenis permainan papan (*board-game)* dengan variasi papan permainan berukuran 3 x 3 yang sekarang telah berkembang menjadi 5 x 5 ataupun 9 x 9. Permainan ini biasanya dimainkan oleh dua orang pemain untuk mengisi waktu luang. Disini penulis tertarik untuk mengembangkan sebuah *game Tic Tac Toe* dengan area yang dinamis yang artinya disini pemain tidak hanya akan bermain pada satu jenis *board* yang ditentukan saja tetapi dapat juga pada jenis papan permainan yang berbeda dengan cara pemain dapat memilih sendiri *board* yang ingin dimainkan, seperti contohnya *board* ukuran 3 x 3, 5 x 5, ataupun 9 x 9 dengan *rule* yang ditentukan. Disini penulis akan mencoba untuk menerapkan algoritma *Memory-Enhanced Test Driver With Value F* pada *Artificial Intelligent (AI)* pemain yaitu teknik yang biasa digunakan untuk memberikan kepintaran buatan untuk lawan tanding yang bukan merupakan pemain. Algoritma *Memory-Enhanced*

Test Driver With Value F berfungsi dalam pemilihan langkah terbaik, sehingga perlawanan AI akan sangat tepat dan sulit untuk melakukan kesalahan. Dalam pembuatan *game* ini, penulis akan menggunakan *framework PhoneGap*. Tujuan dari skripsi ini adalah untuk Menerapkan algoritma *Memory-Enhanced Test Driver With Value F* pada *Artificial Intelligence (AI)* atau kecerdasan buatan dalam pembuatan permainan *Tic Tac Toe* dengan variasi area pada perangkat Android. Dalam implementasi perancangan game ini mempunyai batasan masalah sebagai berikut : (1) Disini penulis menggunakan algoritma *Algoritma Memory-Enhanced Test Driver With Value F* yang diterapkan pada *Artificial Intelligence (AI)* sebagai lawan bermain khususnya pada permainan *Tic Tac Toe*, (2) Pembuatan permainan menggunakan *framework phonegap*, (3) Board permainan menggunakan ukuran jumlah kotak 3 * 3 *(easy)*, 7 * 7 *(normal)*, 9 * 9 *(hard)*, (4) Bidak yang digunakan adalah X dan O. (5) *Game* ini akan diterapkan pada perangkat *Android*. (6) *Game* ini adalah permainan berjenis *single player*.

Metodologi yang digunakan pada aplikasi ini adalah *prototyping*. Prototyping adalah pengembangan yang cepat dan pengujian terhadap model kerja *prototype* dari aplikasi baru melalui proses interaksi dan berulang-ulang yang biasa digunakan ahli sistem informasi dan ahli bisnis.

2. METODE PENELITIAN

2.1 Studi Literatur

2.1.1 Tic Tac Toe

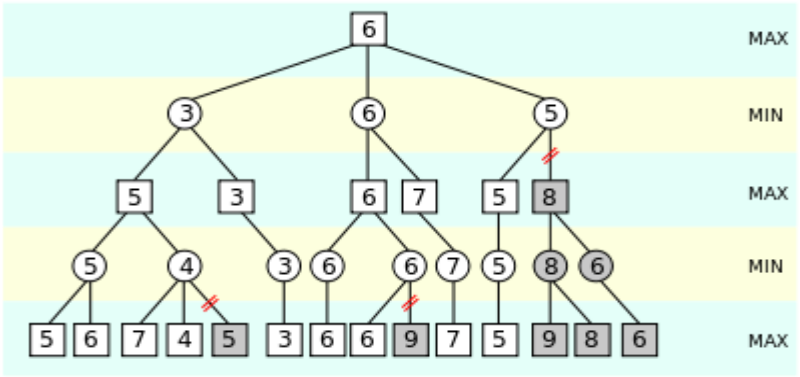
Game Tic Tac Toe merupakan permainan klasik berjenis permainan papan (*board-game)* dengan papan permainan berukuran 3x3 yang sekarang telah berkembang menjadi 5x5 ataupun 9x9. *Game* ini menggunakan dua simbol pemain yaitu X atau O. *Game* ini dimulai dengan mengisi salah satu bentuk simbol pada salah satu bidak, hingga tiga buah simbol yang berbentuk sama tersusun membentuk garis diagonal, vertikal, atau horizontal. *Game* ini biasanya dimainkan oleh dua orang pemain, tapi pada versi *game* berbasis komputer, pemain lawan dapat digantikan oleh komputer.

Dalam *game* ini hasil permainan yang didapat berupa menang, kalah, atau seri. Hasil permainan dinyatakan menang jika simbol dari salah satu pemain (*player* atau komputer) telah tersusun membentuk garis diagonal, vertikal, atau horizontal, sedangkan pemain yang lain dianggap kalah. Hasil permainan dinyatakan seri jika simbol dari kedua permain tidak ada yang tersusun membentuk garis diagonal, vertikal, atau horizontal, serta semua bidak papan permainan telah terisi. Dengan ada nya kecerdasan buatan dalam game ini, maka kesempatan pemain untuk memenangkan permainan jauh lebih sulit. Bahkan kemungkinan terbaik untuk pemain hanyalah seri. Dengan kata lain, dengan menggunakan algoritma, komputer tidak akan kalah

2.1.2 Algoritma MTD (f)

Algoritma *MTD (f)* adalah sebuah algoritma optimasi *Minimax* baru yang lebih sederhana dan lebih optimal daripada beberapa pendahulunya. Nama dari algoritma ini adalah kependekan dari *MTD(,f),* yang disingkat dari Memory-*enhanced Test Driver with node n and value f* (Setiadi, 2012). *MTD* adalah nama dari sekumpulan *driver* program yang mencari pohon *Minimax* menggunakan pemanggilan *Alpha Beta With Memory* berkali-kali dengan menggunakan metode *zerowindow*. Pemanggilan *Alpha Beta With Memory* mengembalikan batas dari nilai evaluasi *Minimax*. Batas dari nilai itu kemudian disimpan dalam *upperbound* (batas atas) dan *lowerbound* (batas bawah), membentuk sebuah interval yang melingkupi nilai *Minimax* yang sebenarnya pada pencarian dengan kedalaman tertentu

Dalam beberapa percobaan permainan komputer seperti Catur, Othello, dan Checkers, algoritma ini mempunyai performa rata-rata lebih baik daripada *Negascout* (variasi dari *Alphabeta* yang diimplementasikan dalam hampir semua permainan catur, checkers, dan othello). Salah satu program catur terkuat, Cilkchess milik MIT yang menggunakan metode komputasi pararel, juga menggunakan *MTD (f)* sebagai algoritma pencariannya menggantikan *Negascout* yang digunakan oleh program catur pendahulunya, *StarSocrates*.



Gambar 1. Pohon *Graph* MTD (f)

Implementasi pada Algoritma MTD (f) terdiri dari 10 baris kode, yaitu seperti berikut:

```
function MTDF(root, f, d)
g := f
upperBound := +•
lowerBound := -•
while lowerBound < upperBound
if g = lowerBound then
• := g+1
else
• := g
g :=
AlphaBetaWithMemory(root, •-1, •, d)
if g < • then
upperBound := g
else
lowerBound := g
return g
```

Gambar 2. *Pseudocode* MTD (f)

Algoritma MTD (f) diatas memanggil fungsi *Alpha Beta With Memory* berkali-kali dengan metode *zerowindow* pencarian *Alpha-beta*, tidak seperti *Negascout* yang menggunakan pencarian *wide-window*.

Efisiensi dari MTD (f) berasal dari pencarian *Alpha-beta* dengan *zero-window*, dan menggunakan sebuah nilai batas yang baik (variabel *beta*) untuk melakukan pencarian *zero-window* tersebut. Dalam versi yang sebelumnya, *Alpha-beta* dipanggil dengan pencarian *wide-window* seperti ini *AlphaBeta* (*root*, *-INFINITY*, *+INFINITY*, *depth*), yang memastikan nilai kembaliannya berada dalam interval *alpha* dan *beta*.

MTD (f) hanya membutuhkan pencarian nol jendela, Test. Bukan dua batas, MTD (f) membutuhkan satu. Dalam NegaScout, ketika nilai-nilai baru untuk jendela pencarian menjadi tersedia mereka harus dikomunikasikan untuk proses di MTD (f). Membatalkan seluruh *subtree* saat *cutoff* sebuah terjadi. Selain itu, rekursif kode pencarian tidak membuahkan kembali pencarian lagi. Semua kembali pencarian dilakukan pada akar, di mana hal-hal yang sederhana daripada turun di pohon paralel. Penelitian tentang Alpha-Beta dan NegaScout dapat langsung diterapkan untuk kasus MTD (f).

Membatalkan seluruh *subtree* saat *cutoff* sebuah terjadi. Selain itu, rekursif kode pencarian tidak membuahkan kembali pencarian lagi. Semua kembali pencarian dilakukan pada akar, di mana hal-hal yang sederhana daripada turun di pohon paralel. Penelitian tentang Alpha-Beta dan NegaScout dapat langsung diterapkan untuk kasus MTD (f), karena mereka menggunakan zero-window Alpha-Beta panggilan untuk melakukan pencarian pohon.

2.1.3 Phonegap

Phonegap adalah sebuah kerangka/framework poen source untuk membuat aplikasi yang dapat dijalankan pada banyak perangkat mobile. *Phonegap* menggunakan bahasa pemrograman web, yaitu HTML, CSS, dan JavaScript sebagai bahasa utama (Andi, h.2). *Phonegap* bekerja dengan cara merubah *web application package* menjadi *native application*. Aplikasi yang telah dibuat akan ditampilkan dalam bentuk *web view* yang memungkinkan pengguna untuk melakukan interaksi dengan aplikasi tersebut. Tampilan dari aplikasi yang dibuat oleh *Phonegap* dibentuk oleh *CSS* yang di implementasikan kedalam *HTML* dan untuk komunikasi data bisa menggunakan *XML* atau *JSON*.

Aplikasi yang telah selesai dibuat dengan *Phonegap* akan dirubah dengan *Software Development Kit (SDK) platform* yang didukung oleh *Phonegap* menjadi *native application.*

Saat ini *PhoneGap* mendukung *platform* terkenal antara lain Apple *iOS, Google Android, HP/Palm webOS, Microsoft Windows Phone 7, Nokia Symbian, RIM BlackBerry, Samsung Bada*. Dikarenakan fungsi setiap *platform* berbeda-beda, maka *Phonegap* menyediakan *Application Programming Interface* (API) untuk mempermudah pengembang diantaranya adalah *Accelerometer, Camera, Capture, Compass, Connection, Contacts, Device, Events, File, Geolocation, Media, Notification,* dan *Storage.* API tersebut dibuat agar pengembang dapat mengakses fungsi *native application* melalui javascript dengan *syntax* yang sama di semua *platform*. Tidak semua *platform* dapat menggunakan API *Phonegap* karena keterbatasan *software* maupun *hardware* tiap *platform*, berikut adalah perbandingan API di masing masing *platform.*

2.1.4 Android SDK

Android adalah system operasi berbasis Linux yang dipergunakan sebagai pengola sumber daya perangkat keras, bak untuk ponsel, smarthphone dan juga PC table. Secara umum Android adalah platform yang terbuka (*Open Source*) bagi para pengembang untuk menciptakan aplikasi sendiri yang digunakan oleh berbagai piranti bergerak. Keunggulan utama Android adalah gratis dan *open source*, yang membuat smartphone Android dijual lebih murah dibandingkan dengan Blackberry atau iPhone meski fitur (*hardware*) yang ditawarkan Android lebih baik.

2.2 Metode Prototyping

Metodologi penelitian yang digunakan dalam penerapan algoritma MTD(f) pada pembuatan pemainan Tic Tac Toe ini adalah metode *prototyping*. *Prototyping* memiliki tujuan untuk menentukan tujuan umum, gambaran bagian-bagian yang dubutuhkan dan kebutuhan yang diketahui. Ada lima fase pada metode *prototyping* yaitu, fase perencanaan, perancangan, evaluasi, membangun sistem, pengujian, dan implementasi.

Prototyping memiliki tahapan-tahapan atau fase yang dapat dilakukan. Berikut penjelasan untuk setiap fase pada *prototyping* :

2.2.1 Perencanaan *Prototyping*

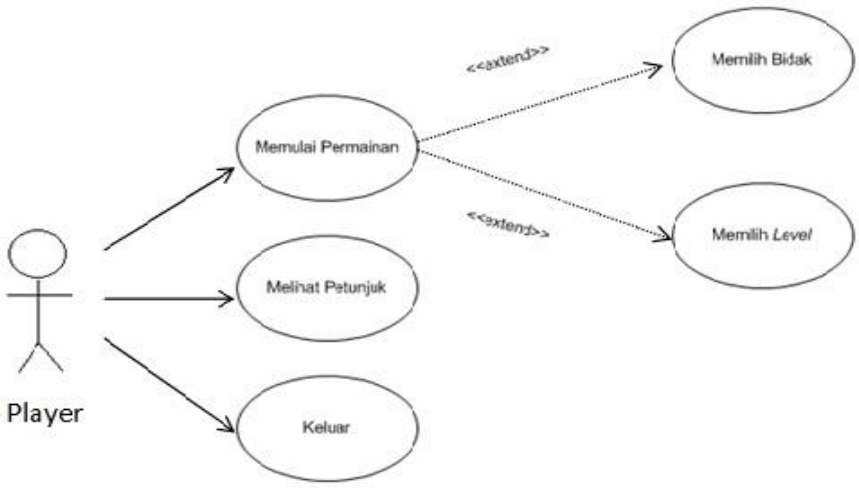
Pada tahap perencanaan ini penulis melakukan identifikasi terhadap kebutuhan permainan, salah satunya melalui studi literature beberapa jurnal penelitian terdahulu.

2.2.2 Mendesain *Prototyping*

Pada tahap desain penulis membuat perancangan sementara yang berfokus pada penyajian aplikasi dan skenario yang akan dibuat sebagai berikut :

2.2.2.1 *Use Case* Aplikasi Permainan

Diagram *use case* sebagai teknik untuk menganalisa kebutuhan utama sistem yang akan dibangun dengan menggambarkan sistem sebagai sekumpulan *use case*, pelaku, serta interaksi antara keduanya dengan sistem. Pada *use case* ini menggambarkan fitur-fitur yang ada dalam permainan dengan *player* adalah sebagai aktor dalam *use case*.



Gambar 3 *Use Case* Aplikasi Permainan

Tabel 1 berikut iniberisi tentang deskripsi – deskripsi mengenai *use case* yang telah dijabarkan sebelumnya serta *actor* yang dapat mengakses *use case* tersebut.

Tabel 1 Glosarium *Use Case* Aplikasi Permainan

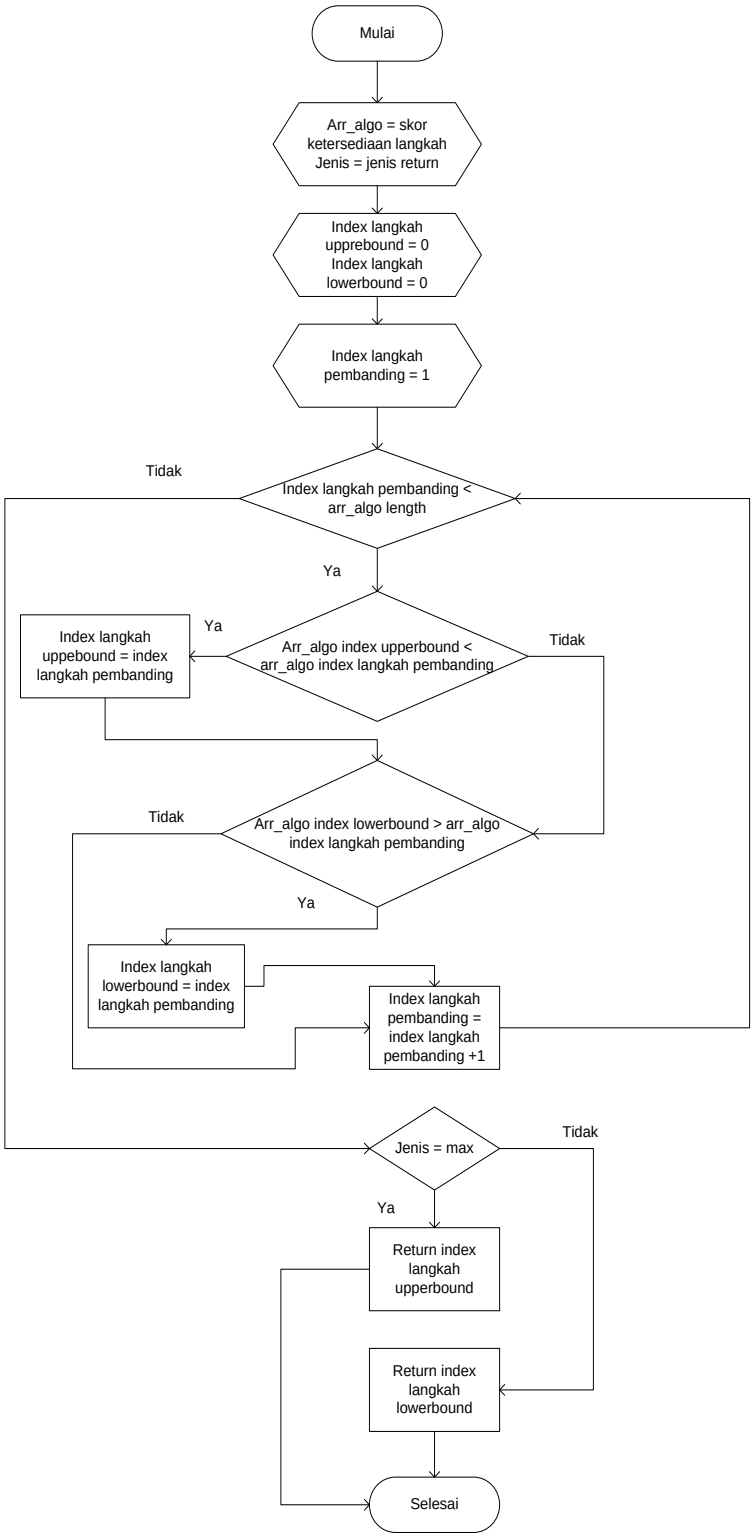
No.	**Nama *Use Case***	**Deskripsi**	**Aktor**
1.	Memulai Permainan	*Use Case* ini digunakan aktor untuk masuk pada submenu memilih bidak dan *level* permainan untuk kemudian memulai bermain.	*Player*
2.	Melihat Petunjuk	*Use Case* ini digunakan aktor untuk melihat petunjuk atau aturan permainan.	*Player*
3.	Keluar	*Use Case* ini digunakan aktor untuk keluar dari aplikasi permainan.	*Player*
4.	Memilih Bidak	*Use Case* ini digunakan untuk memilih bidak yang digunakan untuk bermain.	*Player*
5.	Memilih *Level*	*Use Case* ini digunakan sebelum memulai permainan untuk memilih *level* yang akan dimainkan, *level* yang dimaksud adalah *level easy*, *medium*, dan *hard*.	*Player*

2.2.2.2 Arsitektur Game

Arsitektur *game* bertujuan untuk menjelaskan proses–proses yang terjadi dalam suatu sistem aplikasi. Dalam perancangan alur algoritma dan*game* ini, digunakan alat bantu berupa *flowchart* (diagram alir).

a. Diagram Alir *MTD (f)*

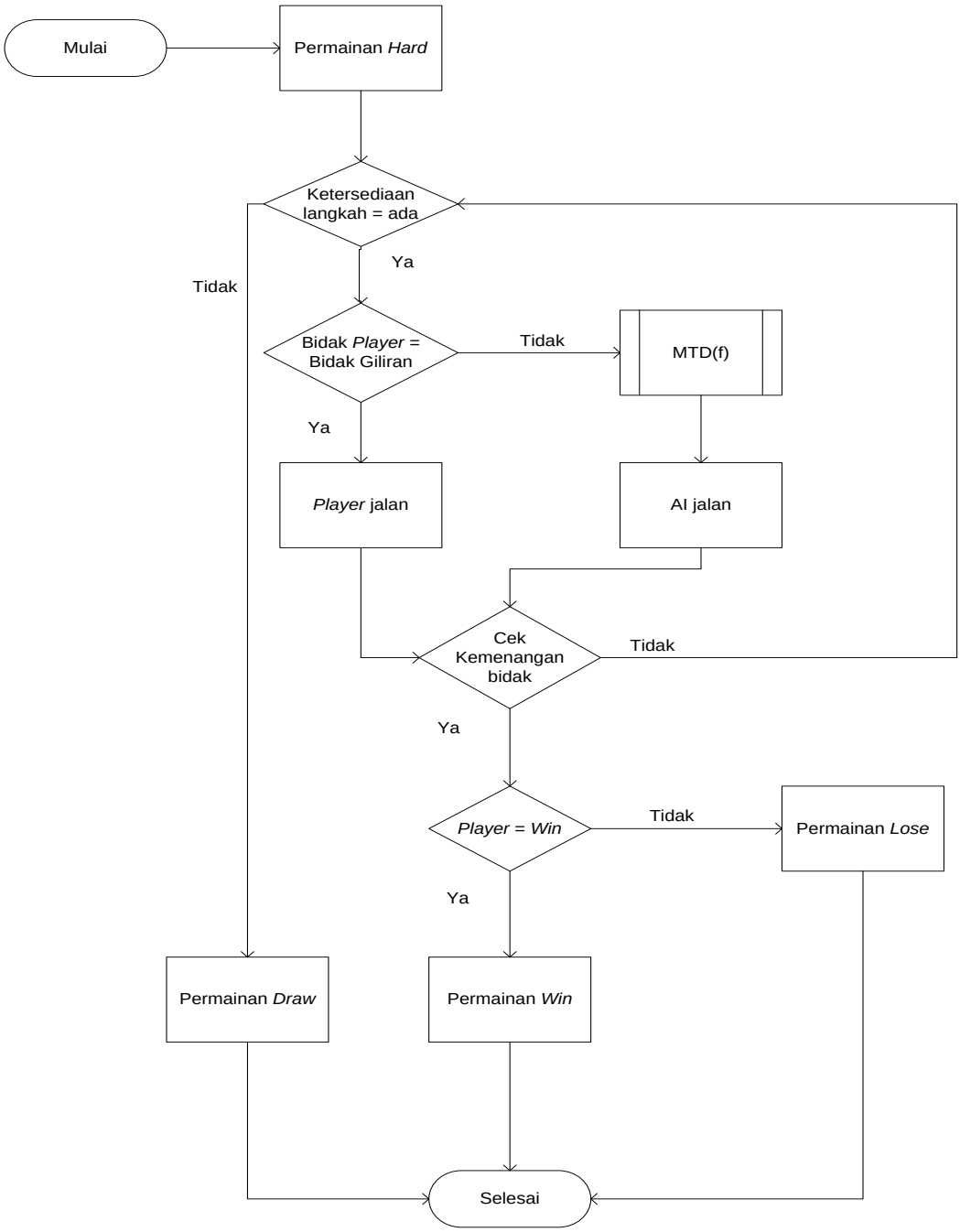
Diagram alir ini akan digunakan apabila sistem menjalankan permainan *level easy* dan *level medium*.



Gambar 4 Diagram Alir *MTD (f)*

b. Diagram Alir *MTD (f)* pada *level hard*

Diagram alir ini akan digunakan apabila sistem menjalankan permainan *level hard*.



Gambar 5 Diagram *Alir MTD (f)* pada *level hard*

2.2.3 Mengevaluasi Prototyping

Pada tahap ini, dilakuan evaluasi terhadap desain *game*, apakah rancangan *game* dan skenario yang dibuat sudah sesuai dengan yang diharapkan. Jika tidak sesuai, maka desain akan direvisi dengan mengulang langkah 2.2.1 dan 2.2.2.

2.2.4 Membangun Sistem

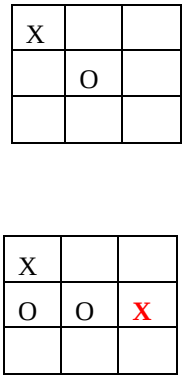
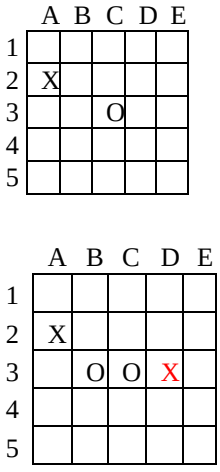
Dalam tahap ini penulis melakukan proses *coding* setelah proses desain dan evaluasi selesai dilakukan yaitu membuat pemain dapat meletakan bidak, bidak AI yang dapat berjalan secara otomatis, dan jenis papan permainan. Pada tahap ini, penulis membangun sistem dengan menggunakan *framework* Phonegap sedangkan bahasa program yang digunakan yaitu HTML, CSS, JavaScprit, dan JQuery. Penulis menerapkan algoritma MTD (F) pada AI agar dapat mengalahkan *player*.

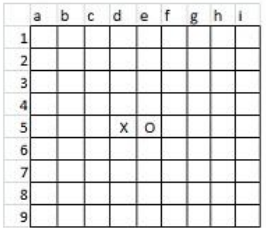
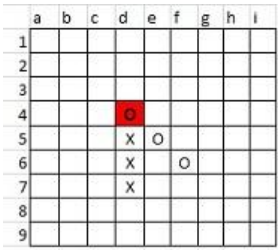
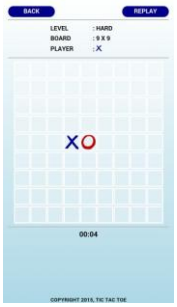
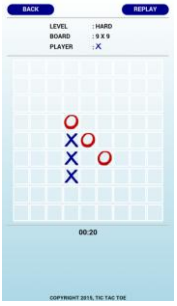
2.2.5 Menguji Sistem

Berdasarkan hasil uji *game* yang telah penulis buat, pengujian implementasi algoritma pada alur permainan dengan metode *black box testing* dapat dilihat pada bagian a adalah tabel 2 pengujian alur permainan dan bagian b adalah alur permainan mengunakan pohon graf dapat dilihat pada gambar 6, 7, 8 adalah pohon graf MTD (f) :

a. Tabel Pengujian Alur Permainan

Tabel 2 Pengujian Alur Permainan

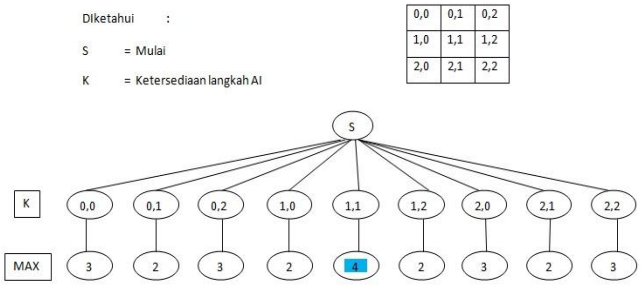
Uji Coba	Penjelasan	Hasil Uji
1.Pengujian Langkah pada *Level Easy*		
	Berikut ini adalah penjelasan AI *Easy* yang digunakan komputer dalam menentukan langkah terbaik, pada pengujian ini asumsi kondisi papan permainan seperti pada gambar disamping, dimana *player* menggunakan bidak X dan AI menggunakan bidak O, AI selalu akan mencari jalan terbaik yaitu disudut papan permainan karena disudut atau di tengah AI mendapatkan 3 peluang jalan yaitu secara horizontal, vertikal, dan diagonal. selanjutnya *player* akan jalan dan bila *player* mendekati syarat untuk menang, pada *level Easy* syarat kemenangan adalah bidak tersusun sejajar secara horizontal, vertikal, dan diagonal maka AI akan menutup jalan *player* seperti pada gambar yang bidak X ditandai dengan warna merah. telah sesuai dan mendapatkan hasil yang terbaik	
2.Pengujian Langkah pada *Level Medium*		
	Berikut ini adalah penjelasan AI *Medium* yang digunakan komputer dalam menentukan langkah terbaik, pada pengujian ini asumsi kondisi papan permainan seperti pada gambar disamping, dimana *player* menggunakan bidak O dan AI menggunakan bidak X, AI selalu akan mencari jalan terbaik yaitu pada papan (A.2) karena AI mendapatkan kesempatan untuk memenuhi syarat dimana pada papan 5x5 syarat yang dibutuhkan untuk menang adalah 4 bidak tersusun secara horizontal, vertikal, dan diagonal. AI memilih jalan di(A.2) karena AI masih dapat jalan secara horizontal di bagian (A.1) secara horizontal. selanjutnya *player* akan jalan dan bila *player* mendekati syarat untuk menang, maka AI akan menutup jalan *player* seperti pada gambar yang bidak X ditandai dengan warna merah	

3.Pengujian Langkah pada *Level Hard*		
 	Berikut ini adalah penjelasan AI *Hard* yang digunakan komputer dalam menentukan langkah terbaik, pada pengujian ini asumsi kondisi papan permainan seperti pada gambar disamping, dimana *player* menggunakan bidak X dan AI menggunakan bidak O, AI selalu akan mencari jalan terbaik yaitu pada papan (E.5) karena AI mendapatkan kesempatan untuk memenuhi syarat dimana pada papan 9x9 syarat yang dibutuhkan untuk menang adalah 5 bidak tersusun secara horizontal, vertikal, dan diagonal. AI memilih jalan di(E.5) karea pada posisi tengah AI mendapatkan jalan terbaik dengan kemungkinan lebih besar pada papan lainnya. selanjutnya *player* akan jalan dan bila *player* mendekati syarat untuk menang, maka AI akan menutup jalan *player* seperti pada gambar dipapan (D.4) yang bidak X ditandai dengan warna merah, selain itu AI mendapatkan bidak tersusun 3 berturut.	 

b. Alur Permainan Mengunakan Pohon Graf

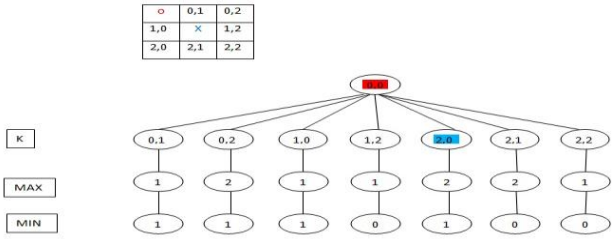
Dalam melakukan pengujian menggunakan pohon graf papan permainan akan dipetakan atau dibuat koordinat untuk menentukan letak langkah yang akan diambil dijelaskan pada Gambar 6, Gambar 7, Gambar 8.

Pada kondisi memulai permainan, algoritma MTD (f) mencari batas tertinggi untuk menambah peluang terbesar memenangkan permainan. disini algoritma memilih langkah yang deberi tanda biru atau koordinat 1,1 dikarenakan memiliki peluang paling besar untuk memenangkan permainan.



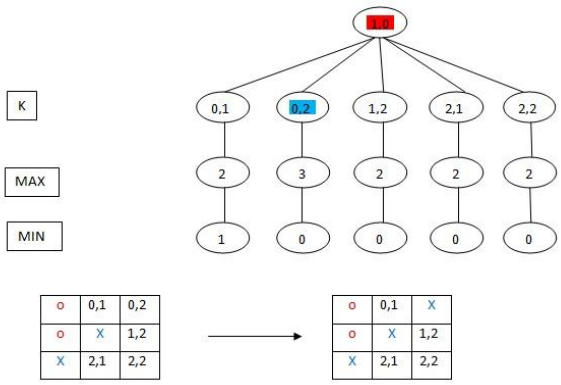
Gambar 6. Pohon Graf MTD (f) Kondisi Awal

Pada gambar 7 player akan memilih langkah terbaik pada koordinat (0,0), setelah itu AI akan memilih koordinat (2,0) karena pada koordinat itu memiliki kemungkinan terbesar dengan nilai 2.



Gambar 7. Pohon Graf MTD (f) Pemilihan Langkah

Pada gambar 8 player akan memilih langkah terbaik pada koordinat (1,0), setelah itu AI akan memilih koordinat (0,2) karena pada koordinat itu memiliki kemungkinan terbesar menang dengan nilai 3.



Gambar 8. Pohon Graf MTD (f) Pemilihan Langkah

2.2.6 Mengimplementasikan Sistem

Dalam tahap implementasi ini, penulis telah membangun sistem yang sesuai dengan kebutuhan melalui proses pengujian yang dianggap telah berhasil dan sesuai. Kemudian penulis melakukan implementasi sistem kedalam *smartphone* agar dapat dimainkan oleh pemain.

3 HASIL DAN PEMBAHASAN

Tahap awal yang harus dilakukan *user* untuk menjalankan aplikasi ini adalah dengan cara mengeksekusi aplikasi sehingga tampil menu utama yang terdiri dari beberapa tombol pilihan, yaitu tombol Play, tombol Instruction, tombol, dan tombol Exit. Pada saat pengguna mengklik tombol Play, maka akan tampil halaman sub menu Play, dimana pengguna dapat memilih warna bidak (O atau X) dan level permainan (easy, medium, atau hard). Saat memilih bidak dan level selesai maka permainan akan dimulai sesuai dengan bidak dan level yang dipilih. Jika pengguna mengklik tombol Instruction pada menu utama maka akan tampil halaman Instruction yang berisi tata cara permainan dan foto membuat. Untuk keluar dari aplikasi pengguna dapat mengklik tombol Exit pada menu utama.

Gambar 9 Tampilan Menu Utama

4 KESIMPULAN

Penarapan algoritma *Memory-Enhanced Test Driver With Value F* pada aplikasi permainan Tic Tac Toe berjalan sesuai dengan alur logika algoritma yang dirancang dan dapat melakukan prediksi langkah selanjutnya serta memberikan solusi langkah dalam permainan sebagai salah satu kemampuan *Artificial Intelligent* (AI). Algoritma *Memory-Enhanced Test Driver With Value F* dapat mengambil langkah dengan kemungkinan menang maksimum bagi komputer dalam permainan serta untuk tingkat kesulitan pada setiap *level* sudah disesuaikan dengan papan bermain. Dari hasil pengujian aplikasi permainan didapatkan bahwa dengan variasi area akan menambah tingkat kesulitan permainan.

5 SARAN

Untuk pengembangan lebih lanjut dan beberapa saran untuk pengembangan aplikasi ke depannya. Ada baiknya jika permainan Tic Tac Toe ini dikembangkan menjadi tampilan 3 dimensi (3D). Ada baiknya jika permainan ini dapat dimainkan oleh 2 pemain. Ada baiknya permainan ini ditambahkan gambar animasi, agar permainan menjadi lebih menarik. Untuk pengembangan permainan selanjutnya ditambahkan fitur pemilihan papan permainan yang bisa diatur sendiri oleh pemain. Program aplikasi permainan Tic Tac Toe dapat dikembangkan menggunakan algoritma pencarian *Heuristic* lainnya.

DAFTAR PUSTAKA

[1] Abdul, K, 2005, *Dasar Pemrograman Web dengan ASP*, Andi Offset, Yogyakarta.

[2] Agung, J, 2015, *Bikin Aplikasi Android dengan Menggunakan Anggulan Mobile Monggo Db*,Loko Media, Jakarta.

[3] *Contoh Pengujian Blackbox, (n.d.)*, Retrieved from, Diakses 11April 2015, dan http://kafegue.com.

[4] EP, J. F, 2007, *Algoritma dan Pemrograman*, Salemba Infotek, Jakarta.

[5] Rizky, R, 2014, *Framework Phonegap*, Retrieved from, Diakses 17Mei 2015, dan http://rororizky.blogspot.com.

[6] Suarga, 2006, *Algoritma Pemrograman*, Andi Offset, Yogyakarta.

[7] Wahana, K, 2013, *Membangun Aplikasi Mobile Cross Platform Dengan Phonegap*, Elex Media, Jakarta.

[8] Khoirush Sholih Ridhwaana Akbar, 2007, *Algoritma Minimax dalam Pengambilan Keputusan pada Permainan Tic-tactoe*, Teknik Informatika ITB, Bandung.

[9] Rainer Feldmann, 2008, *Computer Chess: Algorithm and Heuristics for a Deep Look into The Future*, University of Paderborn, Jerman.

[10] Pohan, H. I., & Bahri, K. S. 1997, *Pengantar Perancangan Sistem*, Erlangga, Jakarta.